Week 6 - Assignment

Programming for Data Science 2024

Exercises for the topics covered in the sixth lecture.

The exercise will be marked as passed if you get **at least 10/15** points.

Exercises must be handed in via **ILIAS** (Homework assignments). Deliver your submission as a compressed file (zip) containing one .py or .ipynb file with all exercises. The name of both the .zip and the .py/.ipynb file **must** be *SurnameName* of the two members of the group. Example: Riccardo Cusinato + Athina Tzovara = *CusinatoRiccardo_TzovaraAthina.zip* .

It's important to use comments to explain your code and show that you're able to take ownership of the exercises and discuss them.

You are not expected to collaborate outside of the group on exercises and submitting other groups' code as your own will result in 0 points.

For questions contact: *riccardo.cusinato@unibe.ch* with the subject: *Programming for Data Science 2024*.

Deadline: 14:00, April 11, 2024.

Exercise 1 - World Happiness Report 8 points

You will find the CSV files *report.csv* and *region.csv* in the data folder for this assignment (Credit: https://www.kaggle.com/ajaypalsinghlo/world-happiness-report-2021).

1. Import the two *csv* files and save them in two variables called *df_report* and *df_region*. Create a new column "region" in the *df_report* dataframe by populating it with the correct region in the *df_region* dataframe. If a country does not exist in *df_region*, label it as "unknown" in *df_report*. Print *df_report* at the end. (*2 points*)

NB If you are unable to do the exercise, you can use the file *report_region.csv* in the data folder for the next points.

In [ ]:
```
###
# YOUR CODE HERE
###
```

2. Calculate and print the median "Healthy life expectancy at birth" per region in the year 2019. (*2 point*)

In [ ]:
```
###
# YOUR CODE HERE
###
```

3. Create a Pivot table with the median "Healthy life expectancy at birth" per region (index) per year (column) and print it. (*1 point*)

In [ ]:
```
###
# YOUR CODE HERE
```

```
###
```

4. Create a Pivot table (same structure as before) with the maximum "Log GDP per capita" per region per year and print it. In the resulting table, also add the overall values across years for each region, with the appropriate pandas method. (*1 point*)

In [ ]:
```
###
# YOUR CODE HERE
###
```

5. Find the length of the shortest country name(s) in the dataset and print it, together with the actual countries (print just the unique occurrences!). Then, create a new column "Short name", where each country name is cut down to the length of the shortest country name. For instance, if the country with the shortest name is Germany (7 letters, and not true, just an example), "Switzerland" would become "Switzer". (*2 points*)

In [ ]:
```
###
# YOUR CODE HERE
###
```

Exercise 2 - Weather data 7 points

In this exercise, you'll use the *weather.csv* dataset that contains UK weather data. The dataset has columns year and month describing which year and month a specific recording belongs to. (Credit: https://www.kaggle.com/josephw20/uk-met-office-weather-data)

1. Import the dataset into a dataframe called *df_weather* and create a new column *datetime*, containing a datetime object for each row describing the year and month of the recording. The day can be set as the 1st of the month. Finally, print the dataframe. (*2 points*)

NB If you are unable to do this exercise, you can use the file *weather_datetime.csv* in the data folder for the next points.

In [ ]:
```
###
# YOUR CODE HERE
###
```

2. Write a function *mean_rainfall* that takes in input the dataframe, the name of a weather station, the upper and lower bounds of a time interval and computes the mean *rain* for the time period between (and including) the upper and lower bound. Make the station name not case-sensitive. Use the inputs as in the example below (i.e. upper and lower bounds as strings) (*3 points*)

```
mean_rainfall(df_weather, "Manston", "january 2019", "march 2020")  # returns 49.52
mean_rainfall(df_weather, "manston", "january 2019", "march 2020")  # also returns 49.52
```

In [ ]:
```
###
# YOUR CODE HERE
###
```

3. Expand the function *mean_rainfall* of the previous point, such that it also returns the number of days since the maximum rainfall in the time-period selected and the current date (use

28th of March as reference) (*2 points*)

```
mean_rainfall(df_weather, "Manston", "january 2019", "march 2020")  # returns 49.52, 1640
```

In [ ]:

```
###
# YOUR CODE HERE
###
```